Symbolic Computing Package for Mathematica (Computer Algebra and Related Topics)

AUTHOR(S):

Yang, Seong-Deog; Chung, Youngjoo

CITATION:

Yang, Seong-Deog ...[et al]. Symbolic Computing Package for Mathematica (Computer Algebra and Related Topics). 数理解析研究所講究録 2017, 2054: 68-76

ISSUE DATE:

2017-10

URL:

http://hdl.handle.net/2433/237148

数理解析研究所講究録
第2054巻 2017年 68-76

Symbolic Computing Package for Mathematica

Seong-Deog Yang*
Dept. of Mathematics, Korea University

Youngjoo Chung†
School of Electrical Engineering and Computer Science
Gwangju Institute of Science and Technology

Abstract

In this article we introduce the Symbolic Computing package for Mathematica, which enables us to manipulate symbolic computations more easily.

1 Motivation of the package

Symbolic Computing Package (SC in short) is a package for Mathematica which has been solely developed by the second author of this article. It extends Mathematica's symbolic computing environment.

The goal of the package is to enable us to manipulate Mathematica's computations as we do with pencil and paper. Why are such goals necessary while Mathematica is supposed to be already one of the best symbolic computing software? Let us point out a few aspects of Mathematica with which we are not happy.

- Input notation problem : There are some mathematical notations we want to use with Mathematica but they are not implemented (yet) in Mathematica. For example, in denoting the derivative of a function f, all the following notations are commonly used in mathematics literature

$$\frac{d}{dx}f(x),\ \frac{df}{dx}(x),\ f',\ \dot{f}$$

but Mathematica do not support them in that meaning at the moment.

- Black Box problem : sometimes we want not only the result but also the process of computations itself. For example, the following shows the result of the integration of $\sqrt{1+x^2}$.

```
In[1]:= Integrate[√(x^2+1),x]

Out[1]= 1/2 (x √(1+x^2)+ArcSinh[x])
```

But we may want, for example, to show to students the actual mathematical algorithm of integration as well as the result.

- Output notation problem : Sometimes the form of outputs of Mathematica computations are not easy to comprehend. We want them to be as close as possible to the notations used in ordinary mathematical literature.

Related to these dissatisfactions, SC provides the following:

*sdyang@korea.ac.kr
†ychung@gist.ac.kr

- new meaning to some existing symbols,
- new symbols,
- more than 800 functions for symbolic manipulation,
- the control over the process of computations using SCMAF, by
  - holding the automatic evaluation of symbols,
  - manipulating them until we desire the evaluation of them,
  - evaluating them.
- the reformatting of the output.

2 Installation of the package

SC is available from `http://symbcomp.gist.ac.kr`. Installation instructions are available for both the personal installation and the system-wide installation. One easiest method of installation is simply to run a one line command

```
In[2]:= ToExpression[URLFetch["http://symbcomp.gist.ac.kr/downloads/InstallSymbCompPersonal.m"]]
```

If the package is successfully installed, you will see two palettes named [SCMAF Viewer] and [Symbolic Computing] under the Palettes menu. The [Symbolic Computing] palette enables us to input various notations corresponding to SC expressions.

After the installation is complete, one can load the package by running

```
In[3]:= <<SymbolicComputing`
```

in a Mathematica notebook.

If you want to uninstall, simply delete the associated files.

If you press 'Help' button in the [Symbolic Computing] palette mentioned above, the [Mathematica help browser] will pop up which contains all the essential information you need to know about in order to be able to use the package. There you can find the name of all the functions defined in the SC. Almost all of them start with 'SC'.

3 Change of the interpretation of notations

The first thing one should bear in mind when using SC is that the usual notations may not work any more as it used without SC. For example, without SC, the following input gets evaluated immediately. However, once you load SC, some notations does not get evaluated any more.

```
In[4]:= ∫ 1/√(4+x^2) dx

Out[4]= ∫ 1/√(4+x^2) dx
```

In order to evaluate the notation, one uses the eval function.

```
In[5]:= SCEvalInt[∫ 1/√(4+x^2) dx]

Out[5]= ArcSinh[x/2]
```

The following is the list of SC functions which are used in evaluating expressions on hold.

```
In[6]:= Names["SCEval*"]

Out[6]= {SCEvalDelta, SCEvalDeriv, SCEvalDot, SCEvalEqSub, SCEvalInt,
  SCEvalIntDelta, SCEvalIntDeltaIfTrue, SCEvalIntDeriv, SCEvalIntIfTrue,
  SCEvalIntResidue, SCEvalLimit, SCEvalMult, SCEvalNInt, SCEvalNProd,
  SCEvalNSum, SCEvalProd, SCEvalProdList, SCEvalSet, SCEvalSum,
  SCEvalSumDelta, SCEvalSumList, SCEvalVecOp}

In[7]:= Length[%]

Out[7]= 22
```

Whenever you find that some notations are not evaluated after the SC is loaded, you may consider applying one of the above eval functions to evaluate them.

Another aspect of the package is that there are some SC symbols which look completely the same as the ordinary symbols. That is, some different expressions are represented by the same notation. Let's look at the following for example.

```
In[8]:= FullForm[{x^2, x^2}]

Out[8]= List[Power[x,2],SCSuperscript[x,2]]

In[9]:= FullForm[{x^-1,x^-1}]

Out[9]= List[Power[x,-1],Inverse[x]]
```

There is a way to distinguish them. If you place the cursor on top of x^2, there may or may not appear a tooltip which shows SCSuperscript[x,2]. If a tooltip does not appear, x^2 is an ordinary Mathematica symbol Power[x,2].

Some new notations are available with SC. For example, Superscripts with more than one index can now be used with SC loaded.

```
In[10]:= SCSuperscript[x,1,2]

Out[10]= x^1,2
```

One can see what SC notations are available by looking at the [Symbolic Computing] palette which can be found at the [Palettes] menu.

With SC, the output of the Mathematica evaluation can take a form which is much closer to the mathematical notation that we see in mathematics literature. For example you will see the following

```
In[11]:= SCDSolve[g''== g,g[x],x, ReplConst→ {c1,c2}]

Out[11]= g[x]==e^x c1+e^-x c2
```

The following example shows that various mathematical notations which denotes the differentiation can be used with SC.

```
In[12]:= f[x_] := Sin[x]

In[13]:= SCEvalDeriv[df[x]/dx]

Out[13]= Cos[x]

In[14]:= SCEvalDeriv[f'[x]]

Out[14]= Cos[x]
```

With x present, x_1 cannot be used, in general, as a constant. But with SC, it can be. See the following example.

```
In[15]:= D[(x+x1)^n,x]

Out[15]= n (x+x1)^(-1+n) (1+Subscript^(1,0)[x,1])

In[16]:= d/dx (x+x1)^n
         SCMAF[%,SCEvalDeriv,All,Hold→x1]

Out[16]= d(x+x1)^n
         ---------
             dx

Out[17]= n (x+x1)^(-1+n)
```

4 SCMAF

Among the more than 800 functions of SC, the function SCMAF is probably the most important function to know. It controls the flow of symbolic computations.

An example of this action is the following, which derives the quadratic formula:

```
In[18]:= a x^2+ b x + c ==0
         SCMAF[%,SCEqMove,{All, -c, Right},
         SCMultEq,{All,a},
         SCAddToEq,{All, b^2/4},
         SCFactor,{At[1], Full → True},
         SCMultEq,{All, 4 },
         RA, {All, (α_^2==β_  → α==±√β) },
         SCSubtFromEq,{All, b},
         SCDivEq,{All,2a},,,
         Together,{At[2]},
         Trace → True]

Out[18]= c+b x+a x^2==0

         b x+a x^2==-c

         a b x+a^2 x^2==-a c

         b^2/4+a b x+a^2 x^2==b^2/4-a c

         1/4 (b+2 a x)^2==b^2/4-a c

         (b+2 a x)^2==b^2-4 a c

         b+2 a x==(±1) √(b^2-4 a c)

         2 a x==-b+(±1) √(b^2-4 a c)

Out[19]= x==(-b+(±1) √(b^2-4 a c))/(2 a)
```

In order to explain how the code works, let us indicate each line in the code by a number in parentheses.

```
(0) a x^2+ b x + c ==0
(1) SCMAF[%,
(2) SCEqMove,{All, -c, Right},
(3) SCMultEq,{All,a},
(4) SCAddToEq,{All,b^2/4},
(5) SCFactor,{At[1], Full → True},
(6) SCMultEq,{All, 4 },
(7) RA, {All, (α_^2==β_ → α==±√β) },
(8) SCSubtFromEq,{All, b},
(9) SCDivEq,{All,2a},,,
(10) Together,{At[2]},
(11) Trace → True]
```

Let us call the result of the evaluation of the line (k) as expression[k]. Expression[1] is by definition $ax^2 + bx + c == 0$. Now expression[1] is passed to the line (2), which produces

```
In[21]:=    SCEqMove[expression[1], -c, Right]
```

and the result of this becomes expression[2]. And so on.

One important aspect of SCMAF's symbolic handling is that it can pinpoint the exact spot in the expression[k] to which a function is to be applied. For example, in line (5), the command which is executed is

```
In[22]:=    SCFactor[expression4[[1]], Full → True]
```

and expression[5] is simply expression[4] with expression[4][[1]] replaced by the result of the above command. So basically we specify the exact spot from an expression to which the commands are to be applied, and then the next expression is obtained.

Another important feature of SCMAF which is not easy to explain by a static article like this is that the execution of the above code is actually very interactive. For example, in the above code, we observe that $\alpha^2 = \beta$ is to be replaced by $\alpha = \pm\sqrt{\beta}$. Well, how do we know that we need to perform this operation at this stage? It is because we build up the entire code while observing the resulting expressions after executing each line.

The way to accomplish this interactive execution of codes involves the use of $ inside SCMAF. It can be put at the end of each command line inside SCMAF, and SCMAF stops its execution after executing that line. Let's look at the following example:

```
In[23]:=  a x^2+ b x + c ==0
          SCMAF[%,SCEqMove,{All, -c, Right},
          SCMultEq,{All,a},$,
          SCAddToEq,{All,b^2/4},
          SCFactor,{At[1], Full → True},
          SCMultEq,{All, 4 },
          RA, {All, (α_^2==β_ → α==±√β) },
          SCSubtFromEq,{All, b},
          SCDivEq,{All,2a},,,
          Together,{At[2]},
          Trace → True]

Out[23]=  c+b x+a x^2==0

          b x+a x^2==-c

Out[24]=  a b x+a^2 x^2==-a c
```

The execution of the code stopped after the execution of line (2). Now we proceed to the next step.

```
In[25]:= a x^2+ b x + c ==0
         SCMAF[%,SCEqMove,{All, -c, Right},
         SCMultEq,{All,a},
         SCAddToEq,{All,b^2/4},
         SCFactor,{At[1], Full → True},
         SCMultEq,{All, 4 },$,
         RA, {All, (α_^2==β_  → α==±√β) },
         SCSubtFromEq,{All, b},
         SCDivEq,{All,2a},,,
         Together,{At[2]},
         Trace → True]

Out[25]= c+b x+a x^2==0

         b x+a x^2==-c

         a b x+a^2 x^2==-a c

         b^2/4+a b x+a^2 x^2==b^2/4-a c

         1/4 (b+2 a x)^2==b^2/4-a c

Out[26]= (b+2 a x)^2==b^2-4 a c
```

The execution stopped after the execution of line (5). Using $, one can observe how the computation is going on, and then one can make intellectual guesses on what the next code should be. For example, with the above code we see that taking the square root of both sides is the next step to go, and that is exactly how we proceeded.

The key point of the all the above is that we can completely control the flow of ideas in the computation, and see all the steps of the computations.

The general usage of SCMAF is as follows:

```
SCMAF[expression[0],
function1, {arguments1, options1}, localOptions1,
function2, {arguments2, options2}, localOptions2,
......,
functionk, {argumentsk, optionsk}, localOptionsk,
......,
functionn, {argumentsn, optionsn}, localOptionsn,
globalOptions
]
```

Roughly speaking, in the first step, function1[expression[0], options1] is executed, and then localOptions1 of SCMAF is applied to produce expression[1]. Then, function2[expression[1], options2] is executed, and the localOptions2 of SCMAF is applied to produce expression[2], so on. Of course, as we see in line (5) and line (10) above, the actual expression to which a function is applied can be only a part of the whole expression.

One of the most useful global options is Trace → True. With it, SCMAF to print out all the expressions that it gets during the execution of the commands inside. With Trace → False, SCMAF prints out only the last two expressions.

There is a convenient way to automate the use of $ in order to control the flow of computations. In the [Palettes] menu, one can find the [SCMAF Viewer] palette. Click any of the circles or arrows in the palette right after evaluating an SCMAF code. For example, click |<< and then >. Then you can observe that the results of the SCMAF is displayed line by line.

5 Working Examples

Now we revisit the problem of integrating $\sqrt{x^2+1}$ which was dealt with in the first section. The problem that we pointed out was that we want to see the process of computations rather than just to get the result. With SC, we can specify how we start by substituting x^2 with $\tan t$:

```
In[27]:= ∫√(x^2+1)dx ;
         SCAFE[%,
         SCTransInt,{All,TransVar→{x,t,x==Tan[t]}}, Apply→{PowerExpand,Simplify}]
```

$$\text{Out[27]=}\quad \int\sqrt{1+x^2}\,dx == \int \text{Sec}[t]^3\,dt$$

Then we proceed as follows to obtain the answer.

```
In[28]:= SCMAF[%,
         SCTransInt,{At[2],TransVar→{t,u,t==ArcSec[u]}},SCMultDenom→{u,Positive→True},
         SCTransInt,{At[2],TransVar→{u,v,u == Cosh[v]}},
         Assuming[v>0, Simplify[#]]&, At[2],
         SCEvalInt,At[2],
         SCTrigToHalf,At[2],
         RA,{At[2], v == ArcCosh[u]},Apply→PowerExpand,RA → √(-1+u) √(1+u) → √(-1+u^2),
         RA,{At[2],u==Sec[t]},RA→t==ArcTan[x],RA → √x^2 == x,
         SCTrigConvert,{ArcCosh[√(1+x^2)],ArcSinh},Apply→PowerExpand,
         Trace→True]
```

$$\text{Out[28]=}\quad \int\sqrt{1+x^2}\,dx == \int \text{Sec}[t]^3\,dt$$

$$\int\sqrt{1+x^2}\,dx == \int \frac{u^2}{\sqrt{-1+u^2}}\,du$$

$$\int\sqrt{1+x^2}\,dx == \int \frac{\text{Cosh}[v]^2\ \text{Sinh}[v]}{\sqrt{-1+\text{Cosh}[v]^2}}\,dv$$

$$\int\sqrt{1+x^2}\,dx == \int \text{Cosh}[v]^2\,dv$$

$$\int\sqrt{1+x^2}\,dx == \frac{v}{2}+\frac{1}{4}\ \text{Sinh}[2\ v]$$

$$\int\sqrt{1+x^2}\,dx == \frac{v}{2}+\frac{1}{2}\ \text{Cosh}[v]\ \text{Sinh}[v]$$

$$\int\sqrt{1+x^2}\,dx == \frac{1}{2}\ u\ \sqrt{-1+u^2}+\frac{\text{ArcCosh}[u]}{2}$$

$$\int\sqrt{1+x^2}\,dx == \frac{1}{2}\ x\ \sqrt{1+x^2}+\frac{1}{2}\ \text{ArcCosh}[\sqrt{1+x^2}]$$

$$\text{Out[29]=}\quad \int\sqrt{1+x^2}\,dx == \frac{1}{2}\ x\ \sqrt{1+x^2}+\frac{\text{ArcSinh}[x]}{2}$$

This capability of SC is sometimes very helpful in educating students. For example, in the next codes, we show that the integral of the function $\frac{1}{\sqrt{4+x^2}}$ can be obtained in two different ways. The first way is to apply the following method of integration.

In[30]:=
```
∫ 1/√(4+x^2) dx
SCMAF[%,
SCTransInt,{All, TransVar → {x,θ,x == 2 Tan[θ]}},
Simplify,All,
PowerExpand,All,
RA,{All, ∫Sec[θ]dθ==Log[Sec[θ]+Tan[θ]]},
SCTrigConvert,{All,Tan},
SCEliminate,{All,x==2 Tan[θ],Tan[θ]},
ExpToTrig,All,
Trace→True]
```

Out[30]=
$$\int \frac{1}{\sqrt{4+x^2}}dx$$

$$2\int \frac{\text{Sec}[\theta]^2}{\sqrt{4+4\ \text{Tan}[\theta]^2}}d\theta$$

$$\int \sqrt{\text{Sec}[\theta]^2}d\theta$$

$$\int \text{Sec}[\theta]d\theta$$

$$\text{Log}[\text{Sec}[\theta]+\text{Tan}[\theta]]$$

$$\text{Log}[\text{Tan}[\theta]+\sqrt{1+\text{Tan}[\theta]^2}]$$

$$\text{Log}[\frac{x}{2}+\sqrt{1+\frac{x^2}{4}}]$$

Out[31]=
$$\text{ArcSinh}[\frac{x}{2}]$$

The second method goes as follows:

In[32]:=
```
<<SymbolicComputing`
```

In[33]:=
```
∫ 1/√(4+x^2) dx
SCMAF[%,
SCTransInt,{All, TransVar → {x,θ,x == 2 Tan[θ]}},
Simplify, 4+4 Tan[θ]^2,,,
PowerExpand,√Sec[θ]^2,
SCEvalInt,All,Apply → {SCMergeLogs,SCTrigToDouble},,,
FullSimplify,At[1],Apply→ {PowerExpand,Expand},
RA,{All, {Tan[θ] == x/2,Sec[θ] == √(1+x^2/4)}},
Trace → True ]
```

Out[33]=
$$\int \frac{1}{\sqrt{4+x^2}}dx$$

$$2\int \frac{\text{Sec}[\theta]^2}{\sqrt{4+4\ \text{Tan}[\theta]^2}}d\theta$$

$$\int \text{Sec}[\theta]d\theta$$

```
Log[Sec[θ]+Tan[θ]]
```

Out[34]= $\text{Log}\left[\frac{x}{2}+\sqrt{1+\frac{x^2}{4}}\right]$

Of course there is nothing wrong with these, since they are the same functions as can be easily checked.

In[35]:= $\text{ArcSinh}\left[\frac{x}{2}\right] == \text{Log}\left[\frac{x}{2}+\sqrt{1+\frac{x^2}{4}}\right]$ // FullSimplify

Out[35]= True

6 Basic mechanism of the SC's notation handling

Roughly speaking, Mathematica interprets symbols in the following way: To start with, it changes a symbol to a box expression. Then it checks if there is a user defined interpretation of the box expression. If there is a user defined interpretation of it, then it is applied. Otherwise, Mathematica's own interpretation is applied.

This mechanism is the key for SC, because by supplying appropriate interpretations of the box expressions, SC can provide new meanings to existing symbols and create new symbols.

7 Summary

Symbolic Computing package for Mathematica

- enables us to use mathematical notation more freely as we like,
- enables us to control the flow of computations, using SCMAF,
- provides more than 800 functions for symbolic computing.

References

[1] `http://symbcomp.gist.ac.kr`

[2] Youngjoo Chung, *Mathematica Help Browser for SC.*

[3] Youngjoo Chung, *Computer-based Mathematical Analysis Using Mathematica and Symbolic Computing Package*, Mathematica Notebook for DahanTech Special Lecture, 2013. 7. 18.

[4] 양성덕, 이장훈, 황지원, 이채영, 안명환, 기본에 충실한 *Mathematica* 입문, 교우사, 2014.